Comparing Numerical Solution Methods for the Cahn-Hilliard Equation[1]

Riley Fisher[2], Mckenzie Garcia[3], Nagaprasad Rudrapatna[4]

Abstract

This paper considers schemes based on the convexity splitting technique – linear extrapolation, second-order backward difference formulas, and implicit-explicit Runge-Kutta methods – for solving the Cahn-Hilliard equation, a model for phase separation, with periodic boundary conditions. The Cahn-Hilliard equation, which is derived from a gradient flow of an energy functional, is a higher-order, parabolic nonlinear partial differential equation. As such, each of the proposed solvers preserve the energy decreasing property of this equation. The solution methods, which were implemented by incorporating fast Fourier transformations, were compared by means of accuracy and computational runtime. Furthermore, the convexity splitting parameter, a, was varied in order to observe its effects on the accuracy and stability of numerical solvers. The best method for modest accuracy was determined to be the first-order linear extrapolation method. The preferred high-order solver was an implicit-explicit Runge-Kutta scheme, which had a relatively low computational cost. Simulations confirmed that this method preserves stability at larger timesteps while maintaining a high degree of accuracy at both large and small timesteps.

Keywords: *Cahn-Hilliard equation, convexity splitting, fast Fourier transformations, implicit-explicit Runge-Kutta, energy decreasing property*

[1] Prof. Saulo Orizaga, Department of Mathematics, New Mexico Tech, Socorro, NM 87801 (saulo.orizaga@nmt.edu)
[2] Duke University, Durham, NC 27708 (riley.fisher@duke.edu)
[3] Duke University, Durham, NC 27708 (mckenzie.garcia@duke.edu)
[4] Duke University, Durham, NC 27708 (nagaprasad.rudrapatna@duke.edu)

Contents

1 Introduction

Various numerical solvers for the Cahn-Hilliard (CH) equation are discussed in this paper. The CH equation is a higher-order, parabolic nonlinear partial differential equation (PDE),

$$u_t = -\epsilon^2 \Delta^2 u + \Delta(u^3 - u) \tag{1}$$

where Δ is the Laplacian operator, u is related to material concentration, and ϵ represents the thickness of a layer separating the materials. This equation is derived from a gradient flow of an energy functional; thus, solutions must evolve into configurations such that the energy strictly decreases over time. Therefore, numerical methods of this PDE that retain this energy decreasing property (energetic stability) are preferred.

Several energy-stable approaches have been implemented to numerically solve the CH equation. These include convexity splitting, invariant energy quadratization, and the more recent scalar auxiliary variable (SAV) methods. The SAV approach reformulates the CH equation with an auxiliary variable that depends on energy properties and allows for efficient implementation [10]. The line of investigation presented in this paper considers the convexity splitting technique, initially proposed by Elliott and Stuart [3] and later refined by Eyre [4] for the CH equation. More specifically, the numerical solvers that will be analyzed are linear extrapolation (LINX) and second-order backward difference formulas (BDF) – explored by Glasner and Orizaga [5] – and implicit-explicit (IMEX) Runge-Kutta methods – formulated by Song [11] – for the CH equation with periodic boundary conditions.

1.1 Applications

When this equation was first studied by Cahn and Hilliard [2] in 1958, mathematicians and materials scientists were very interested in modeling phase separation phenomena in metallurgy. Accordingly, the CH equation can accurately model the separation of a binary alloy (e.g. an iron-nickel alloy) into its constituent metals. Today, this equation and its many extensions can precisely describe phase separation behavior in a wide array of disciplines, including physics, image processing, ecology, and medicine.

The CH equation, modified by the Navier-Stokes equation, can model a two-phase fluid flow. Similar to the separation of a binary alloy, the movement of two immiscible fluids (e.g. oil and water) can be modeled as they segregate over time [6, 7]. Another extension of this PDE can accurately model two-dimensional microstructure evolution with strong elastic inhomogeneity. This shows the deformation of an object of this type when a force is applied to it [7].

The CH equation can also be adapted for the fast inpainting of binary images, in which missing or damaged portions of these images are filled in using information from surrounding areas. Referencing the color density of non-damaged regions, this CH extension can restore a damaged image to mimic its original appearance [1, 7].

The phase separation principle relies on a density-dependent relation; in the ecological context, this relation translates to animals adjusting their movement directly or indirectly based on the behavior of conspecifics. As such, an extension of this equation describes the movement-driven, spatial self-organization of animals, where species tend to disperse at low and very high densities but aggregate at intermediate density. Such patterned clumping has been observed in a number of animals, including mussels and ants [8].

More recently, potentially life-saving extensions of the CH equation have been proposed. In particular, this PDE can be modified to model tumor growth in human bodies. Whereas the original CH equation involves two distinct materials, this altered CH model designates tumor and healthy tissue as its two phases. This adapted model can aid medical professionals in monitoring and predicting the growth of tumors over time [7, 13].

2 Problem Formulation

2.1 Derivation

We begin by considering that the CH equation is defined as a gradient flow,

$$u_t = \Delta \frac{\delta F}{\delta u} \tag{2}$$

where $\frac{\delta F}{\delta u}$ is defined as the first variation of the energy, $F(u)$:

$$F(u) = \iint_{\Omega} \frac{\epsilon^2}{2} \mid \nabla u \mid^2 + \frac{u^4}{4} - \frac{u^2}{2} \, dx \, dy. \tag{3}$$

Using calculus of variations, the following expression for the first variation of the energy is obtained,

$$\frac{\delta F}{\delta u} = -\epsilon^2 \Delta u + (u^3 - u). \tag{4}$$

Thus, by Equation 2,

$$u_t = \Delta(-\epsilon^2 \Delta u + u^3 - u). \tag{5}$$

By distribution, Equation 1 is achieved.

2.2 Convexity Splitting

By applying the CS method to Equation 3, the following is yielded:

$$F(u) = \iint_{\Omega} \frac{\epsilon^2}{2} \mid \nabla u \mid^2 + \frac{au^2}{2} + \frac{u^4}{4} - \frac{(1+a)u^2}{2} \, dx \, dy. \tag{6}$$

In Equation 6, the convexity splitting parameter, a, is introduced. See Section 2.2.1 for the classical condition on a. $F(u)$ can now be understood as the sum of two distinct portions,

$$F(u) = \iint_{\Omega} F_+ \, dx \, dy + \iint_{\Omega} F_- \, dx \, dy \tag{7}$$

where F_+ represents the convex portion,

$$F_+ = \frac{\epsilon^2}{2} \mid \nabla u \mid^2 + \frac{au^2}{2}, \tag{8}$$

and F_- represents the concave portion of the energy equation,

$$F_- = \frac{u^4}{4} - \frac{(1+a)u^2}{2}. \tag{9}$$

Equation 2 can then be interpreted in a similar manner,

$$u_t = \Delta \frac{\delta F}{\delta u} = \Delta \frac{\delta F_+}{\delta u} + \Delta \frac{\delta F_-}{\delta u}, \tag{10}$$

where

$$\frac{\delta F_+}{\delta u} = -\epsilon^2 \Delta u + au \tag{11}$$

and

$$\frac{\delta F_-}{\delta u} = u^3 - (1+a)u. \tag{12}$$

Therefore

$$u_t = \Delta(-\epsilon^2 \Delta u + au) + \Delta(u^3 - (1+a)u). \tag{13}$$

2.2.1 The Classical Condition On The Convexity Splitting Parameter

The convexity splitting parameter, a, must be greater than two to preserve the energy's convex and concave parts [4]. Recall the concave portion of the energy, F_-, given in Equation 9. In order to retain concavity, $F_-'' < 0$. Taking the second derivative,

$$F_-'' = 3u^2 - (1+a) < 0. \tag{14}$$

The set boundaries for material concentration are $-1 \leq u \leq 1$, with $u = -1$ representing a pure concentration of material A and $u = 1$ representing a pure concentration of material B. By using either extreme value of u,

$$F_-'' = 3(1) - (1+a) < 0. \tag{15}$$

Hence, it is proved that $a > 2$ in order to maintain concavity. Analytically, this condition must be obeyed; however, to observe its effects on the accuracy and stability of numerical solvers, a was varied. This analysis is given in Section 4.4.

2.3 Fast Fourier Transformations

We can derive the CS method proposed by Eyre [4] from Equation 13:

$$\frac{u^{n+1} - u^n}{\Delta t} = -\epsilon^2 \Delta^2 u^{n+1} + a\Delta u^{n+1} + \Delta((u^n)^3 - (1+a)u^n). \tag{16}$$

Eyre proposed that the linear portion of the PDE be evaluated at the next time level due to its predictable behavior, and the nonlinear portion be evaluated at the current time level, to avoid nonlinear systems of equations. In this research, we found that by applying fast Fourier transformations, the efficiency of this technique can be significantly increased [12]. While the CS method, as formulated in [4], uses finite difference methods to form tridiagonal matrices, we instead propose the incorporation of fast Fourier transformations. This approach allows for inverses to be computed very efficiently, which is typically a computationally expensive procedure without fast Fourier transformations [12]. Hence, this implementation increases the efficiency at each timestep.

Applying the Fourier transformation[5]:

$$\frac{\hat{u}^{n+1} - \hat{u}^n}{\Delta t} = -\epsilon^2 \widehat{\Delta^2 u^{n+1}} + a\widehat{\Delta u^{n+1}} + \widehat{\Delta((u^n)^3 - (1+a)u^n)}. \tag{17}$$

Recall that $\widehat{\Delta u} = -k^2\hat{u}$ and $\widehat{\Delta^2 u} = k^4\hat{u}$, where k represents the wave number:

$$\frac{\hat{u}^{n+1} - \hat{u}^n}{\Delta t} = -\epsilon^2 k^4 \hat{u}^{n+1} - ak^2\hat{u}^{n+1} - k^2(\widehat{(u^n)^3} - (1+a)\hat{u}^n). \tag{18}$$

Solving the equation for $\hat{u}^{n+1}$:

$$\hat{u}^{n+1} = \hat{u}^n + \Delta t(-\epsilon^2 k^4 - ak^2)\hat{u}^{n+1} - \Delta t k^2(\widehat{(u^n)^3} - (1+a)\hat{u}^n) \tag{19}$$

$$(1 + \epsilon^2 k^4 \Delta t + ak^2\Delta t)\hat{u}^{n+1} = \hat{u}^n - \Delta t k^2(\widehat{(u^n)^3} - (1+a)\hat{u}^n) \tag{20}$$

$$\hat{u}^{n+1} = \frac{\hat{u}^n - \Delta t k^2(\widehat{(u^n)^3} - (1+a)\hat{u}^n)}{1 + \epsilon^2 k^4 \Delta t + ak^2 \Delta t}. \tag{21}$$

Letting $h = \Delta t$:

$$\hat{u}^{n+1} = \frac{\hat{u}^n - hk^2[(\widehat{(u^n)^3} - (1+a)\hat{u}^n]}{[1 + h(\epsilon^2 k^4 + ak^2)]}. \tag{22}$$

The inverse fast Fourier transformation is then applied to Equation 22 to obtain the value of u^{n+1}.

3 Other Candidate Methods

Prior research suggests that the CS method is not sufficiently accurate [5, 11]. We confirm this assertion in our error plots in Section 4.3. As such, we investigate other numerical methods, including linear extrapolation, second-order backward difference formulas, and implicit-explicit Runge-Kutta schemes.

3.1 Linear Extrapolation

The linear extrapolation (LINX) method, an approach proposed by Glasner and Orizaga [5], is similar to the CS method in formulation. The objective of the LINX method is to produce smaller errors in the implementation of the CS scheme by constructing an iterated initial condition and applying a simple linear extrapolation. This extrapolation, displayed in Equation 23, is used to improve the previous time stepping method and is then iterated a fixed number of times for each timestep. We selected iteration values of $n_{it} = 1, 2, 4, 8$.

$$u^* = 2u^n - u^{n-1} \tag{23}$$

[5] Hat notation is used to denote transformed quantities by convention

3.2 Backward Difference Formulation

In this paper, the backward difference formulation (BDF) refers to the second-order backward difference formulas, as outlined below and in Glasner and Orizaga [5]. As with the LINX method, an improved initial condition is formulated. Compared to LINX, a more precise estimation for the time derivative of u, as seen on the left-hand side of Equation 24, is used. This coupled with a second-order extrapolation technique, shown in Equation 25, gives a higher order of accuracy when compared against the CS and LINX methods. The iteration technique and n_{it} values used for LINX are mimicked for the BDF extrapolation.

$$\frac{\frac{3}{2}u^{n+1} - 2u^n + \frac{1}{2}u^{n-1}}{\Delta t} = -\epsilon^2\Delta^2 u^{n+1} + a\Delta u^{n+1} + \Delta((u^n)^3 - (1+a)u^n) \tag{24}$$

$$u^* = 3u^n - 3u^{n-1} + u^{n-2} \tag{25}$$

3.3 Implicit-Explicit Runge-Kutta

The implicit-explicit (IMEX) Runge-Kutta methods were applied by Song [11] to the CH equation. We used the second-order, three-stage scheme, outlined below in Equations 26-28:

$$u^{(1)} = u^n + \Delta t\big(-\epsilon^2\Delta^2 u^{(1)} + a\Delta u^{(1)} + \Delta((u^n)^3 - (1+a)u^n)\big) \tag{26}$$

$$u^{(2)} = \alpha_{10}u^n + \alpha_{11}u^{(1)} + \beta_1\Delta t\big(-\epsilon^2\Delta^2 u^{(2)} + a\Delta u^{(2)} + \Delta((u^{(1)})^3 - (1+a)u^{(1)})\big) \tag{27}$$

$$u^{n+1} = \alpha_{20}u^n + \alpha_{21}u^{(1)} + \alpha_{22}u^{(2)} + \beta_2\Delta t\big(-\epsilon^2\Delta^2 u^{n+1} + a\Delta u^{n+1} + \Delta((u^{(2)})^3 - (1+a)u^{(2)})\big). \tag{28}$$

However, unlike in Song's original approach, fast Fourier transformations were applied to avoid the intricacies of matrix algebra. The transformed second-order, three-stage scheme is shown below in Equations 29-31:

$$\hat{u}^{(1)} = \hat{u}^n + \Delta t\big(-\epsilon^2k^4\hat{u}^{(1)} - ak^2\hat{u}^{(1)} - k^2(\widehat{(u^n)^3} - (1+a)\hat{u}^n)\big) \tag{29}$$

$$\hat{u}^{(2)} = \alpha_{10}\hat{u}^n + \alpha_{11}\hat{u}^{(1)} + \beta_1\Delta t\big(-\epsilon^2k^4\hat{u}^{(2)} - ak^2\hat{u}^{(2)} - k^2(\widehat{(u^{(1)})^3} - (1+a)\hat{u}^{(1)})\big) \tag{30}$$

$$\hat{u}^{n+1} = \alpha_{20}\hat{u}^n + \alpha_{21}\hat{u}^{(1)} + \alpha_{22}\hat{u}^{(2)} + \beta_2\Delta t\big(-\epsilon^2k^4\hat{u}^{n+1} - ak^2\hat{u}^{n+1} - k^2(\widehat{(u^{(2)})^3} - (1+a)\hat{u}^{(2)})\big). \tag{31}$$

Song [11] stipulated four different sets of α and β values as there is no unique solution for their corresponding system of equations. Table 1 lists the four sets of α and β values. Song assumed that each set of values would produce equally accurate results. This assumption is further explored in Section 4.3.

Scheme	α_{10}	α_{11}	α_{20}	α_{21}	α_{22}	β_1	β_2
2.4	$\frac{3}{2}$	-$\frac{1}{2}$	0	0	1	$\frac{1}{2}$	1
2.5	2	-1	$\frac{1}{2}$	0	$\frac{1}{2}$	1	1
2.6	2	-1	0	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{2}$
2.7	$\frac{5}{2}$	-$\frac{3}{2}$	$\frac{2}{3}$	0	$\frac{1}{3}$	$\frac{3}{2}$	1

Table 1: Parameters For IMEX Runge-Kutta

4 Discussion

4.1 Reference Solution

As shown in the derivation of the CH equation, this PDE contains nonlinear terms. This means that an analytical/closed-form solution cannot be found. We instead used our most accurate method when working with small timesteps, BDF, to create a solution that mimics the theoretical answer. Using the $n_{it} = 4$ BDF scheme on a 256×256 grid ($N = 256$) with spatial domain $[0, 2\pi]^2$, $\epsilon = 0.1$, a convexity splitting parameter of $a = 2$, a final time of 3.2 seconds, and a timestep of 10^{-6} seconds, we constructed a solution that is sufficiently accurate for our analysis. Our chosen initial condition, slightly evolved from a random set of data, and our constructed reference solution are displayed in Figure 1. Due to our limited research time, this was the level of accuracy we settled for as computational costs became too high if we increased the number of iterations or decreased the timestep.

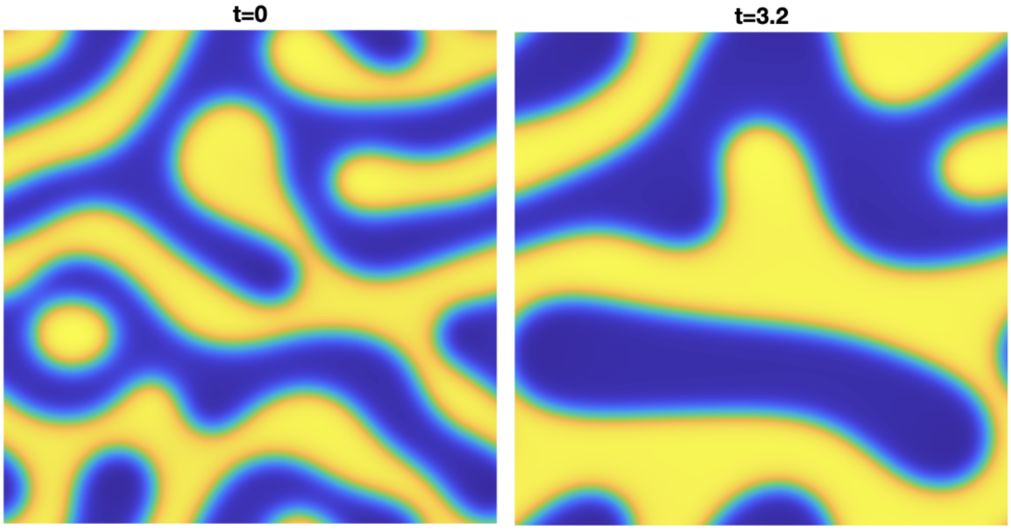


Figure 1: A fine timestep of $10^{-6}s$ was used to generate the reference solution (domain: $[0, 2\pi]^2$, $\epsilon = 0.1$, $N = 256$). Left: The initial condition used in our tests is shown. Right: The reference solution used in our accuracy assessments is shown.

We considered each of the candidate schemes under various criteria to determine a preferred method. All tests were simulated to a final time of 3.2 seconds on a 256 × 256 ($N = 256$) grid with spatial domain $[0, 2\pi]^2$ and $\epsilon = 0.1$, using the initial condition displayed above in Figure 1. These criteria included: the time it took for each *MATLAB* algorithm to run on an Intel Core i5 CPU @2.4 GHz, the accuracy of each scheme relative to the reference solution formulated above, and the effects of varying the convexity splitting parameter on the accuracy and stability of numerical solvers.

4.2 Computational Costs

Using a modest timestep of 0.001 seconds, a convexity splitting parameter of $a = 2.5$, and a final time of 3.2 seconds, we ran each scheme's *MATLAB* code. Table 2 lists the runtime for each scheme. Details about the accuracy of each scheme are provided in Section 4.3, but in general, more accurate schemes took longer to run. For example, the runtime of the LINX or BDF schemes approximately doubled when the number of iterations doubled. Overall, the BDF schemes had longer runtimes than the respective LINX schemes, which is consistent with the higher complexity of the schemes. This trade-off between accuracy and computational runtime is to be expected, particularly when using a modest timestep.

Scheme	Runtime (s)	Scheme	Runtime (s)
CS	3.079288	Song (2.5)	13.48775
LINX $n_{it} = 1$	3.751850	BDF $n_{it} = 1$	4.800847
LINX $n_{it} = 2$	6.516647	BDF $n_{it} = 2$	9.010402
LINX $n_{it} = 4$	11.687746	BDF $n_{it} = 4$	18.676221
LINX $n_{it} = 8$	23.107965	BDF $n_{it} = 8$	34.531473

Table 2: Runtimes with $a = 2.5$, $dt = 0.001s$, $t_f = 3.2s$

4.3 Accuracy

Approximate solutions from each scheme were compared with the reference solution over a range of timesteps using the L^1 norm detailed in Equation 32, where Ω represents the spatial domain and h denotes the spatial discretization step size. We decided to use a convexity splitting parameter of $a = 2.5$ in our analysis to limit the significant instability seen in previous investigations of these methods [5]. A set of timesteps from the order of $10^{-3}s$ to $10^{-1}s$ were used to give a robust understanding of the behavior of each scheme under such conditions.

$$Error = \iint_{\Omega} \mid u_{approximation} - u_{reference} \mid \, dx\, dy \approx h^2 \sum_{i=0}^{n-1} \mid u_{approximation}(x_i, y_i) - u_{reference}(x_i, y_i) \mid \quad (32)$$

The error associated with the CS scheme was first plotted and used as a baseline for each subsequent method. The error associated with each LINX scheme, plotted in Figure 2, shows that linear extrapolation of the CS scheme increases accuracy with little change in stability, even with larger timesteps. Given a particular level of accuracy, we are interested in the LINX scheme with the maximum timestep. In this case, the $n_{it} = 1$ LINX method satisfies this condition and, due to the relatively small difference in stability and accuracy between iterations, we chose the $n_{it} = 1$ LINX scheme for our final comparison.

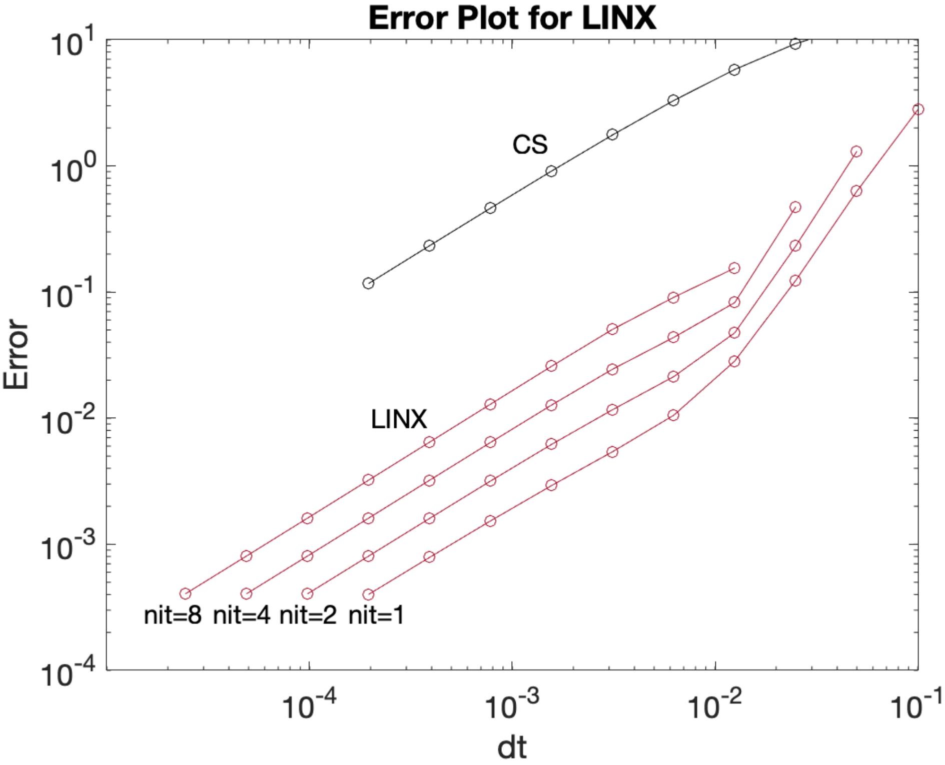


Figure 2: An error vs. timestep plot for the LINX method was constructed with iterations of $n_{it} = 1, 2, 4, 8$. The LINX schemes are shown in red and the CS scheme is shown in black for comparison.

Next, we analyzed the IMEX Runge-Kutta methods by assessing the assumption that each of the four sets of α and β values stipulated by Song yield equally accurate solutions [11]. While examining the accuracy and stability of the second-order, three-stage scheme outlined in Section 3.3 with each of the four sets of α and β values, it became clear that this assumption was unfounded. The error plot below, Figure 3, confirms that the Song (2.5) values resulted in the greatest accuracy, especially when working with smaller timesteps. Accordingly, this set of α and β values was used for our final comparison and later analysis.

The final error plot, Figure 4, displays the errors associated with the chosen LINX and Runge-Kutta schemes, along with the errors corresponding to the CS and BDF schemes. In accordance with Glasner and Orizaga [5], we discovered numerical instability with the BDF methods. However, our chosen parameter of $a = 2.5$ subdued a portion of the instability observed by Glasner and Orizaga, who selected a parameter of $a = 1.5$ [5]. The consequences of varying this parameter are further discussed in Section 4.4. Instability can be particularly noted at the larger timesteps of the $n_{it} = 1$ BDF scheme, but all iterations of this method are afflicted with some instability as the timestep increases.

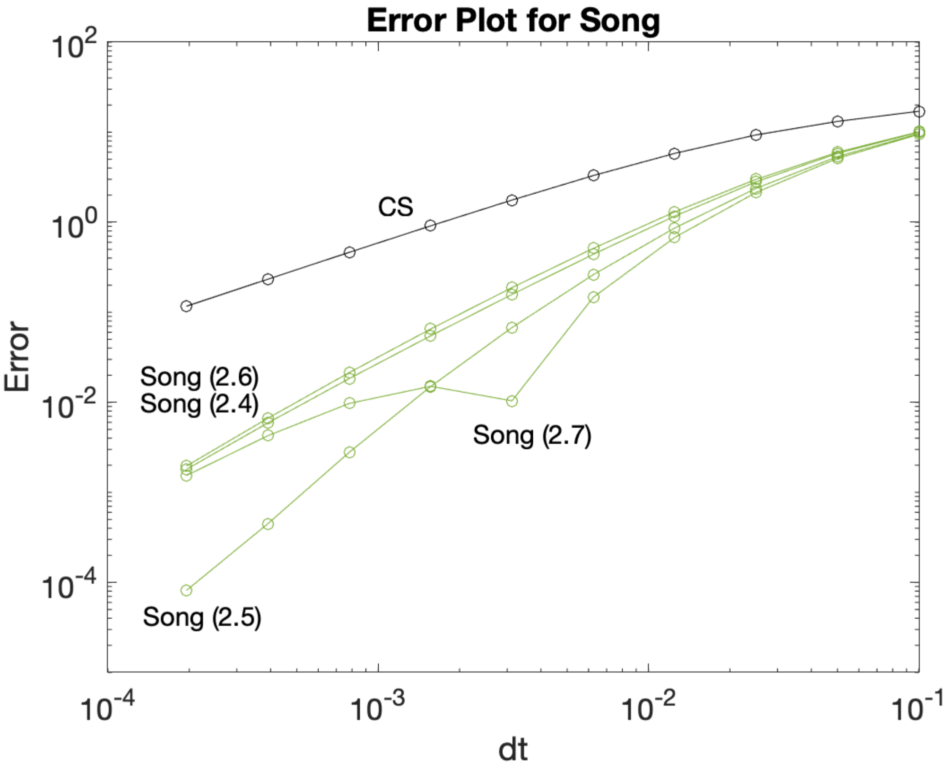


Figure 3: An error vs. timestep plot for the IMEX Runge-Kutta method was constructed for each set of α and β values determined by Song. The Runge-Kutta schemes appear in green and the CS scheme is displayed in black for comparison.

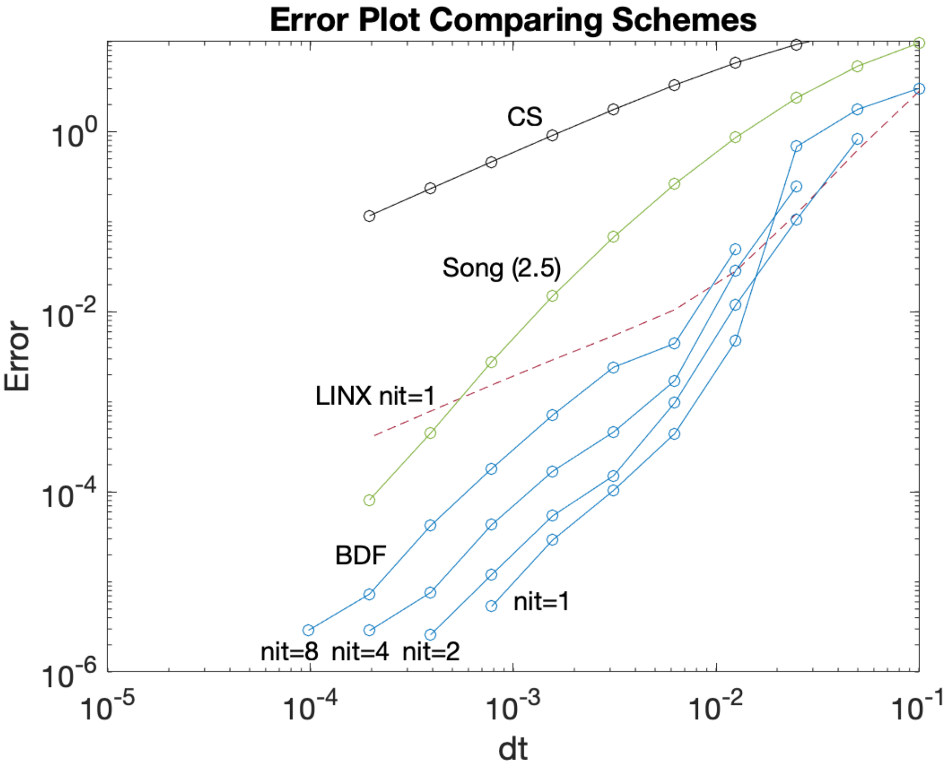
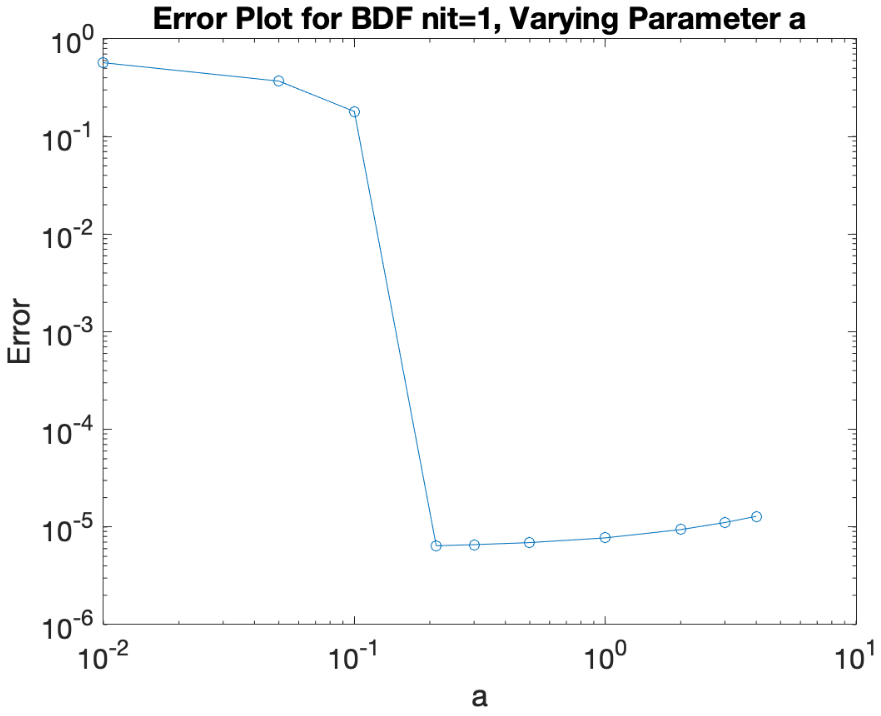


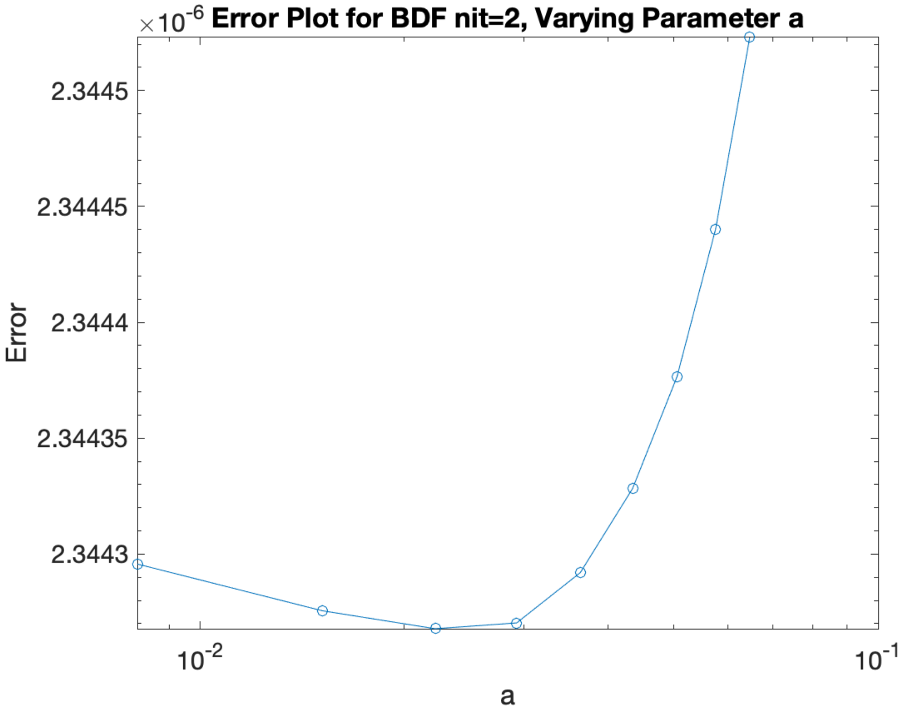
Figure 4: An error vs. timestep plot for the BDF method was constructed with iterations of $n_{it} = 1, 2, 4, 8$. The BDF schemes are shown in blue. The $n_{it} = 1$ LINX scheme is shown in red, the IMEX Runge-Kutta method with Song (2.5) values is shown in green, and the CS scheme is shown in black for a full comparison of the four methods.

Still, the $n_{it} = 4$ BDF scheme seems to be the most useful of the four, as it minimized the risks associated with this method. Its instability was subdued by the increased number of iterations (decreased timestep). Moreover, the $n_{it} = 4$ BDF scheme was not encumbered by as heavy of a computational cost as the $n_{it} = 8$ BDF scheme was (see Section 4.2 for a complete discussion of computational costs). This balance of sufficient stability and plausible runtime suggests the $n_{it} = 4$ BDF scheme to be most desirable. In general, the BDF method suffered increased instability and decreased accuracy with larger timesteps – an issue that did not arise in other methods.

4.4 Varying Parameter *a*

The current literature regarding these numerical solution methods of the CH equation has largely neglected the effects of varying the convexity splitting parameter (deviating from the classical condition discussed in Section 2.2.1). Figure 5 illustrates the effects of varying this parameter for the $n_{it} = 1$ and $n_{it} = 2$ BDF schemes and the IMEX Runge-Kutta scheme with Song (2.5) values.





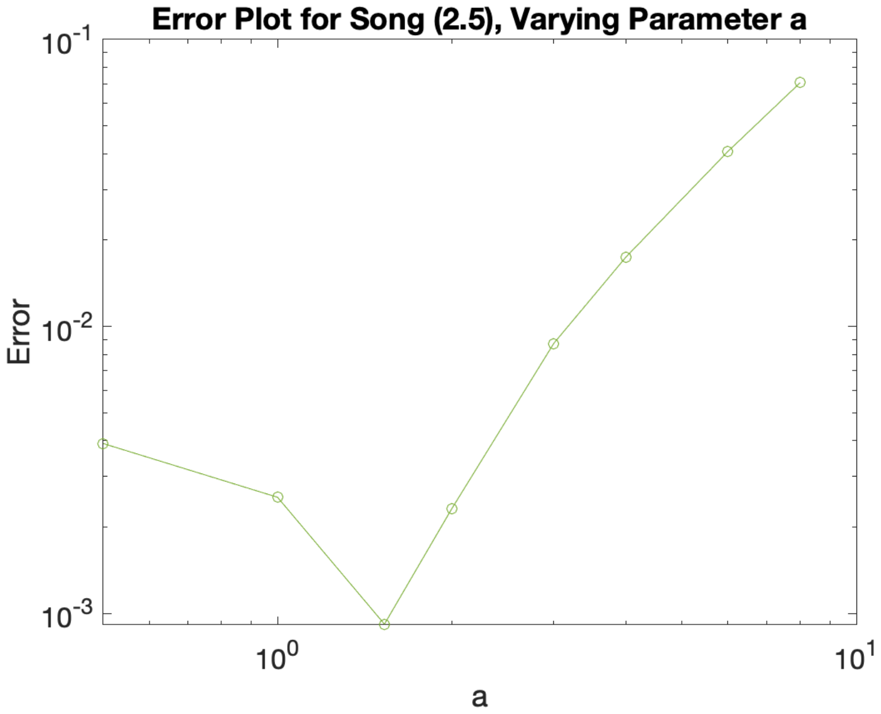


Figure 5: Error vs. parameter a plots were constructed in the hopes of finding the specific a values that minimized the error for each numerical solution method. A fixed timestep of $0.001s$ was used for each plot. Top: The error vs. parameter a plot for the $n_{it} = 1$ BDF scheme is displayed in blue. Middle: The error vs. parameter a plot for the $n_{it} = 2$ BDF scheme is displayed in blue. Bottom: The error vs. parameter a plot for the IMEX Runge-Kutta scheme with Song (2.5) values is displayed in green.

These second-order methods were more likely to show effects when varying a than their first-order counterparts. The $n_{it} = 1$ BDF method and IMEX Runge-Kutta scheme with Song (2.5) values showed the greatest changes in error (over orders of magnitude). The $n_{it} = 2$ BDF scheme showed a slight change in error but not as drastic as the other two. The change in error is over one order of magnitude, so this scheme does not have a significant dependence on the parameter a, as with the other two (see Figure 5). Nevertheless, each of the three schemes had a specific convexity splitting parameter value that minimized its error. These are provided in Table 3.

Scheme	a value
$n_{it} = 1$ BDF	0.212
$n_{it} = 2$ BDF	0.015
IMEX Runge-Kutta	1.53

Table 3: Approximate a Values That Produce Minimum Error

Such behavior was not observed in the CS scheme nor the LINX schemes. We also did not observe a specific a value in the $n_{it} = 4$ and $n_{it} = 8$ BDF schemes that minimized their errors. This is could be due to the increased stability of this type of scheme over higher iterations.

5 Conclusion

5.1 Proposed Method

Taking into account the criteria discussed above – computational runtime, accuracy over various timesteps, and accuracy when varying the parameter a – we concluded that the IMEX Runge-Kutta scheme with Song (2.5) values was most desirable. This numerical solution method preserves stability at larger timesteps while maintaining a high degree of accuracy at large and small timesteps. Additionally, this method requires fewer computational resources than the $n_{it} = 4$ BDF scheme (see Table 2). It is important to reiterate that the $n_{it} = 4$ BDF scheme is the best BDF method due to its subdued instability, compared to its $n_{it} = 1$ and $n_{it} = 2$

counterparts, and decent runtime, compared to the $n_{it} = 8$ BDF scheme. Although the BDF method is slightly more accurate than the IMEX Runge-Kutta scheme, the IMEX Runge-Kutta method is preferred due to its increased stability and computational efficiency. If modest accuracy is all that is desired, we also concluded that the first-order LINX method suffices.

Figure 6 demonstrates our proposed Runge-Kutta method in action over longer periods of time, using a timestep of 0.001 seconds. These snapshots highlight the long-term behavior of the CH equation, particularly how two materials eventually segregate into two distinct regions.

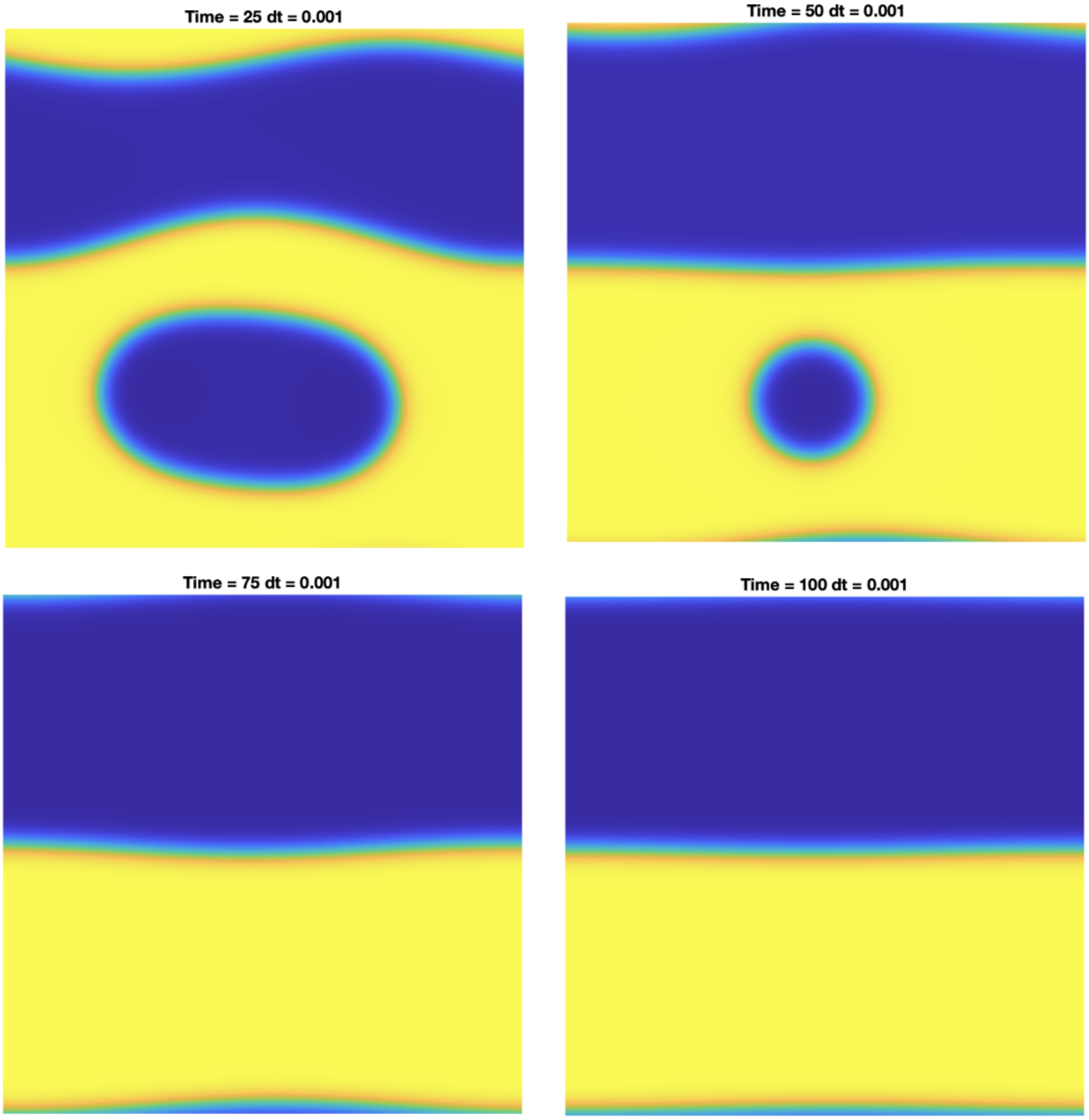


Figure 6: The proposed IMEX Runge-Kutta scheme was run over longer periods of time to give a visual of this method in action ($dt = 0.001s$, domain: $[0, 2\pi]^2$, $\epsilon = 0.1$, $N = 256$). From top left to bottom right: IMEX Runge-Kutta scheme with Song (2.5) values at $t = 25s$, $t = 50s$, $t = 75s$, $t = 100s$.

Figure 7 records the energy of this method over time. As required by the CH equation for any numerical solution method, the energy always decreases. The maintenance of energetic stability (energy decreasing property) justifies our choice of the preferred IMEX Runge-Kutta scheme.

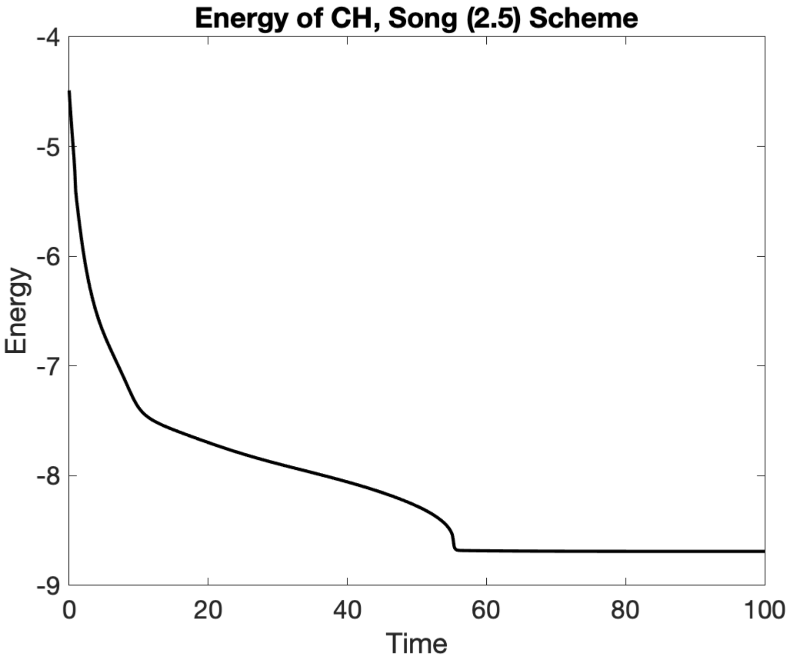
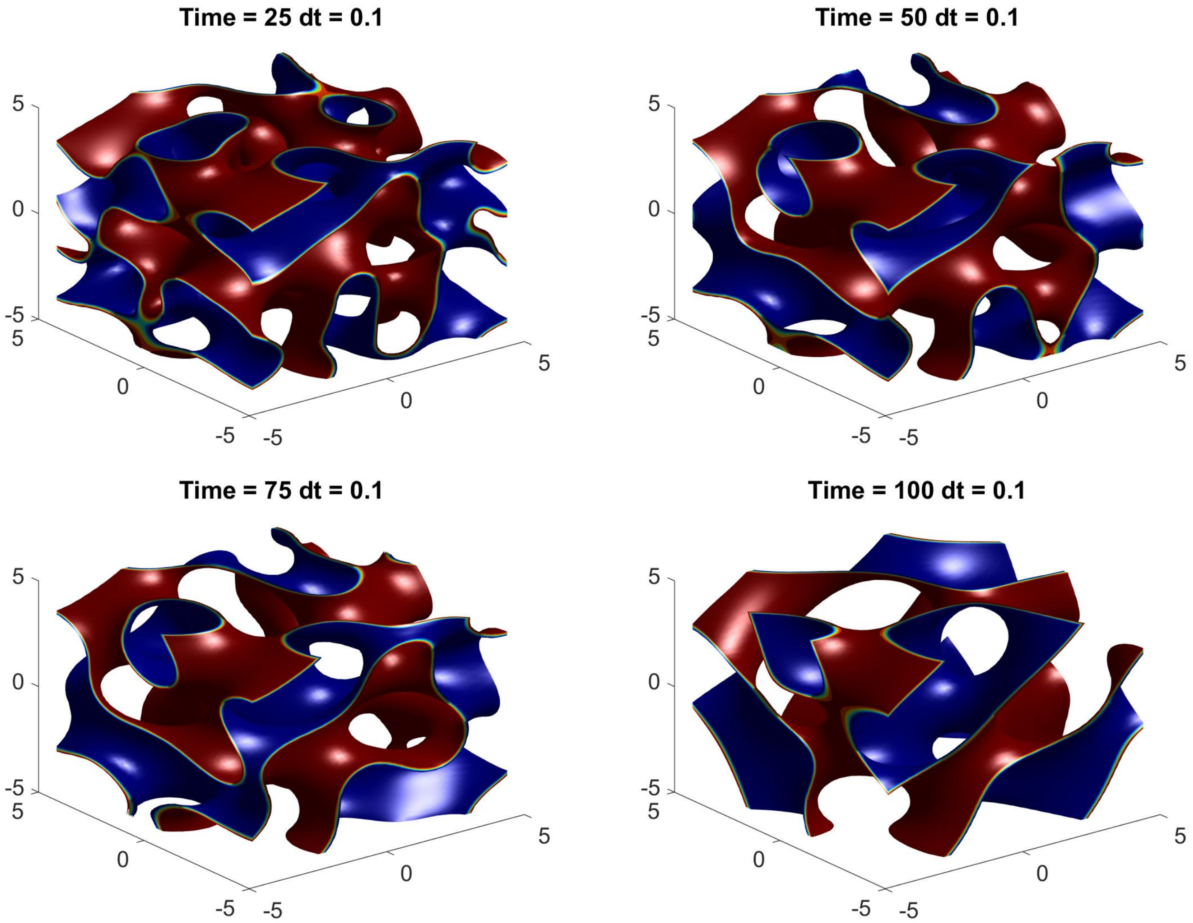


Figure 7: Energy was plotted over time for the preferred method ($dt = 0.001s$, domain: $[0, 2\pi]^2$, $\epsilon = 0.1$, $N = 256$). The energy decreasing property is upheld as energy is strictly decreasing.

5.2 Extension to Three Dimensions

This proposed method provides numerical solutions to the two-dimensional CH equation. As many applications require the three-dimensional version of this PDE, we adapted our preferred method, the IMEX Runge-Kutta scheme with Song (2.5) values, to three-dimensional space numerics, as evidenced by Figure 8.



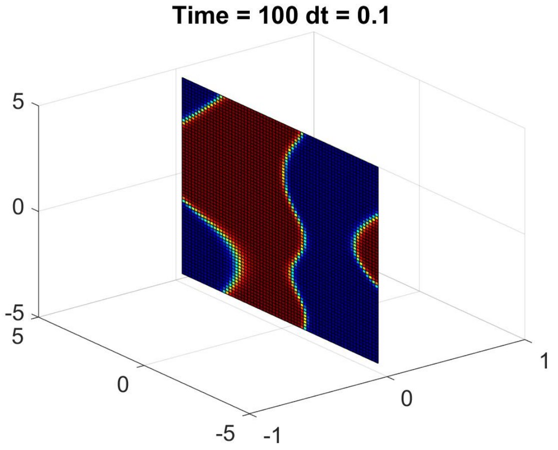


Figure 8: The proposed IMEX Runge-Kutta method is compatible with 3-dimensional space numerics ($dt = 0.1s$, domain: $[0, 2\pi]^3$, $\epsilon = 0.1$, $N = 64$). From top left to middle right: 3D IMEX Runge-Kutta scheme with Song (2.5) values at $t = 25s$, $t = 50s$, $t = 75s$, $t = 100s$. Bottom: Cross section of 3D IMEX Runge-Kutta scheme with Song (2.5) values at $t = 100s$.

6 Future Research

In the future, we would like to explore extensions of the CH equation, including Cahn-Hilliard systems and the block copolymer equation. We primarily worked with the two-dimensional CH equation, so it would be interesting to work more extensively with the three-dimensional version, perhaps while modeling tumor growth or other biological phenomena.

Solutions to the block copolymer equation can be much more fascinating than solutions to the CH equation because the block copolymers – which are composed of long sequences of the same monomer unit, covalently bound to sequences of unlike type – do not completely separate. This process, known as micro-segregation, results in the development of strange patterns. For example, gyroid and spherical patterns are possible depending on the evolution of the block copolymers over time with temperature fluctuations [9].

Along with investigating these variants of the CH equation, we would like to increase the accuracy of our solutions by utilizing third-order numerical methods. Historically, instability has proven to be a formidable barrier for researchers; however, a similar implementation of fast Fourier transformations with appropriate splitting and time stepping techniques could provide a significant breakthrough.

7 Acknowledgments

The authors would like to thank the Department of Mathematics at Duke University for sponsoring this research and the DOmath program as a whole. In particular, the authors recognize Professors Lenhard Ng and Heekyoung Hahn for their involvement in organizing DOmath 2020. The authors are especially grateful to their sponsor, Professor Saulo Orizaga, and project manager, Lihan Wang, for their indispensable guidance and mentorship throughout the process.

References

[1] A. L. Bertozzi, S. Esedoglu, and A. Gillette, *Inpainting of binary images using the Cahn–Hilliard equation*, IEEE Trans. Image Process., 16 (2007), pp. 285–291.

[2] J. W. Cahn and J. E. Hilliard, *Free energy of a nonuniform system. I. interfacial free energy*, The Journal of Chemical Physics, 28 (1958), pp. 258–267.

[3] C. M. Elliott and A. M. Stuart, *The global dynamics of discrete semilinear parabolic equations*, SIAM Journal on Numerical Analysis, 30 (1993), pp. 1622–1663.

[4] D. J. Eyre, *Unconditionally gradient stable time marching the Cahn-Hilliard equation*, MRS Proceedings, 529 (1998), pp. 39–46.

[5] K. Glasner and S. Orizaga, *Improving the accuracy of convexity splitting methods for gradient flow equations*, J. Comput. Phys., 315 (2016), pp. 52–64.

[6] D. Kay and R. Welford, *Efficient numerical solution of Cahn–Hilliard–Navier–Stokes fluids in 2D*, SIAM J. Sci. Comput., 29 (2007), pp. 2241–2257.

[7] J. Kim, S. Lee, Y. Choi, S.-M. Lee, and D. Jeong, *Basic principles and practical applications of the Cahn–Hilliard equation*, Math. Probl. Eng., (2016).

[8] Q.-X. Liu, M. Rietkerk, P. M. Herman, T. Piersma, J. M. Fryxell, and J. van de Koppel, *Phase separation driven by density-dependent movement: a novel mechanism for ecological patterns*, Physics of Life Reviews, 19 (2016), pp. 107–121.

[9] M. W. Matsen and F. S. Bates, *Block copolymer microstructures in the intermediate-segregation regime*, The Journal of Chemical Physics, 106 (1997), pp. 2436–2448.

[10] J. Shen, J. Xu, and J. Yang, *A new class of efficient and robust energy stable schemes for gradient flows*, SIAM Review, 61 (2019), pp. 474–506.

[11] H. Song, *Energy SSP-IMEX Runge-Kutta methods for the Cahn-Hilliard equation*, J. Comput. Appl. Math., 292 (2015), pp. 576–590.

[12] L. N. Trefethen, *Spectral Methods in Matlab*, SIAM, Philadelphia, 2001.

[13] X. Wu, G. van Zwieten, and K. G. van der Zee, *Stabilized second-order convex splitting schemes for Cahn–Hilliard models with application to diffuse-interface tumor-growth models*, Int. J. Numer. Methods Biomed. Eng., 30 (2014), pp. 180–203.